社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

Transformerを用いたクロスモーダルレシピ検索・画像生成

楊　　景†　柳井　啓司††

† 電気通信大学 情報理工学域 I 類
†† 電気通信大学 大学院情報理工学研究科 情報学専攻
E-mail: †yang-j@mm.inf.uec.ac.jp, ††yanai@cs.uec.ac.jp

あらまし　インターネットの発展により，マルチモーダルデータが指数的に増大している．その中でも，食物は人間の生活で重要な役割を果たしているため，料理レシピサイトに多数存在している，料理画像と調理レシピテキストからなるマルチモーダルデータの研究が近年注目されている．特に，画像からテキスト，テキストから画像の相互検索を行うクロスモーダルレシピ検索は，多くの研究が行われている．本研究は，当研究室で行われた研究で，現在，最高精度のクロスモーダルレシピ検索を実現している RDEGAN [16] の啓発の下で，Transformer を用いたレシピ検索フレームワークを提案する．実験では，提案手法はレシピ検索・画像生成の二つのタスクにおいてともに従来の手法を上回った結果を得られたことが示された．

キーワード　Transformer，クロスモーダル検索，自己教師あり学習，画像生成

1. はじめに

近年インターネットの発展により，膨大な量のデータが人は利用できるようになった．この膨大な量のデータを利用すべく，クロスモーダル検索の研究が盛んでいる．クロスモーダル検索の目的は，異なる情報源からの関連のある情報の，互いの検索を可能とすることである．クロスモーダル検索は通常の検索とは違い，明確かつ一意的対応関係がないため，良い精度の検索システムの開発は一般的には困難である．クロスモーダル検索を実現する手法の一つとしては，二つのモダリティからの情報をそれぞれのエンコーダーを通してから，ネットワークでその対応関係を学習することである．異なるモダリティの特徴を，同じ空間に埋め込み，なるべく分布を近づけることで，画像・テキストの互いの検索が可能となった [7].

クロスモーダル検索における応用タスクの一つとして，レシピ検索が挙げられる．レシピのクロスモーダルレシピ検索の目的は，与えられたクエリー画像に対して対応するレシピテキスト，逆にクエリーレシピテキストに対して対応するレシピ画像を，それぞれ検索結果として返すことである．多くのレシピ検索の研究において，Recipe1M [14] がデータセットとして使用されている．

Recipe1M は，レシピテキストとその対応のレシピ画像を含むデータセットである．レシピの方は，タイトル・成分・調理手順が含まれている．同時に，それぞれのレシピに対応するレシピ画像が付いている．

ただし，クロスモーダル検索におけるレシピ検索タスクの問題点の一つとして挙げられることは，似ているレシピテキスト・レシピ料理画像の間の区別が困難なことである．レシピ画像が似ているが，レシピテキストは全く違うサンプル同士や，レシピテキストが似ているが，レシピ画像が全く違うサンプル同士の区別はレシピ検索システムには難しいのである．本研究は，上記の問題点を解決すべく，Hierarchical Transformer (H-T) [13] と Adversarial Cross-Modal Embedding (ACME) [18] を組み合わせたフレームワークを提案する．レシピテキストのエンコードに自己教師あり学習を導入するほか，GAN [5] を画像を再構成することで，現存手法を上回る現時点での最高精度を達成したことが実験で確認された．

2. 関連研究

2.1 Joint Embedding によるレシピ検索

クロスモーダルレシピ検索の一つの発端として，Salvador らの研究 [14] があげられる．この研究で Salvador らは，Recipe1M レシピデータセットを提示すると同時に，クロスモーダルエンベディングによるレシピ検索の手法 (Joint Embedding) を提案した．Salvador らは，レシピ画像・レシピテキストをそれぞれのネットワークと通して特徴ベクトルを生成する．この二つの特徴ベクトルを共有空間に埋め込む同時に，互いの分布を近づけさせることで，レシピ画像・レシピテキストの互いの検索を可能にした．

2.2 画像生成が伴うレシピ検索

Joint Embedding (JE) [14] のフレームワークをさらに発展させ，敵対的学習を導入したことで検索を可能にしたほか，レシピテキストからの画像生成を可能としたのが R2GAN [20] である．この敵対的学習のフレームワークを Wang らがさらに発展させ，一致性損失関数などの導入で高い検索精度を実現した手法 ACME [18] を提案した．ACME はレシピエンベディングを用いた画像再構成に GAN [5] を導入するほか，画像エンベディングを用いた成分予測などを導入し，画像の中間表現のエンベディングの信頼性を上げた．本研究のレシピ検索システムは，ACME をベースのアーキテクチャーとする．

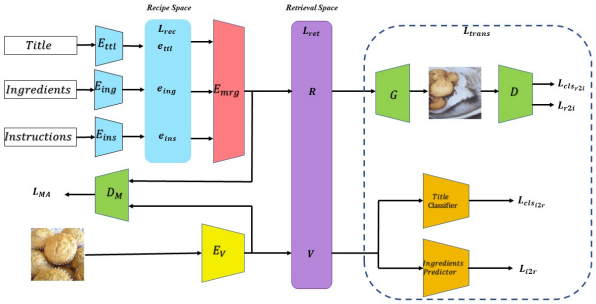

図 1　提案ネットワーク図

同じレシピであっても，料理の盛りつけの仕方や料理写真の撮影の方法によって様々な料理写真が考えられるため，レシピエンベディングから直接，料理画像を生成する ACME には問題があると杉山らは考え，この問題点を対策とした ACME をさらに発展させた手法 RDEGAN [16] を提案した．RDEGAN は，MUNIT [10] の啓発で ACME のネットワークを改良した上で，レシピ画像から画像意味情報と皿や盛り付け方などの情報を含む形状情報を別々にして抽出し，検索精度・画像生成の二つにタスクにおいて当時最高精度を達成した．

2.3　クロスモーダル検索における Transformer の利用

近年 Transformer [17] というテキストエンコーダーの新しいネットワークが提案され，テキスト処理の多くのタスクにおいて良いパフォーマンスを達成している．Recipe1M データセット作成者でもある Salvador らは，データセットは 7 割近くのデータはレシピテキストしか含まない問題点を提示し，この問題点に対する対策を考慮したクロスモーダルレシピ検索手法 [13] を提案した．この方法では，テキストエンコーダーに階層的 Transformer を導入する同時に，自己教師あり学習も適用し，レシピテキストの間にある補完関係をより良く利用した．本研究では，この Transformer に基づくテキストエンコーダーを利用することとする．

3.　提 案 手 法

本研究では，画像生成を行うマルチモーダルレシピ検索のモデルである ACME [18] に，Salvador らの階層的 Transformer 構造を用いたレシピ検索フレームワーク [13] (以下では，本研究を H-T と記載する) で用いられている Transformer 型のテキストエンコーダと自己教師学習を組み合わせ，さらに画像エンコーダに Vision Transformer (ViT) [3] を利用することを提案する．図 1 は，本研究が提案したネットワーク図である．

H-T [13] と同様に，本研究はレシピテキストにある補完的関係を探るために自己教師あり学習のトリプレットロス $!_{A42}$ を適用する．画像エンコーダーは，ResNet50 [8] や ViT [3] を利用する．検索精度を上げるため，学習されたテキスト埋め込み・画像埋め込みに検索用のトリプレットロス $!_{A4C}$ を適用した距離学習を行う．また，モダリティ間のギャップ問題を緩和するために，我々は ACME [18] と同様に損失関数 $!_{"}$ を導入する．同時に，上記の過程で学習された埋め込みが正しく本来のレシピ情報を保持するために，エンベディングの一致性を確保する損失関数 $!_{CA0=B}$ を導入する．以上の点に従って，全体的ロス関数は以下のように定式化できる．$_{-1}$-$_{-2}$-$_{-3}$ は学習の度合いを制御するためのハイパーパラメータである．

$$!_{C>C0;} = _{-1}!_{A42} + _{-2}!_{"} \quad + _{-3}!_{CA0=B} + !_{A4C} \tag{1}$$

3.1　テキストエンコーダー

より信頼性のあるレシピエンベディングを得るために，本研究ではテキストエンコーダーは H-T [13] のテキストエンコーダーを利用する．

レシピテキスト埋め込みを学習するため，レシピテキストに含まれる三つの要素ータイトル，成分と調理手順をそれぞれ階層的 Transformer のエンコーダー $_{CC;}$- $_{8="}$- $_{8=B}$ で先にエンコードし，タイトル，成分と調理手順それぞれのエンベディング $4_{CC;}$-$4_{8="}$-$4_{8=B}$ を得る．その後，この三つエンベディング $4_{CC;}$-$4_{8="}$-$4_{8=B}$ を連結して，レシピテキストエンコーダー $_{A42}$ でエンコードし，最終のレシピテキストエンベディングを得る．

ここでは，レシピテキストに含まれる，タイトル，成分と調理手順の間の補完的意味を掘るために，[13] と同様に損失関数の自己教師あり損失関数を導入する．自己教師あり損失関数の導入により，Recipe1M [14] に含まれる 7 割弱の text-only サンプルを学習に利用できる [13].

3.1.1　タイトルのエンコーディング

すべてのレシピテキストにおいて，タイトルは一つに定まっているため，タイトルに対するエンコードは実際一つの文に対するエンコードである．我々は二つの Transformer を，文レベルのエンコーディングに利用する (図 2(a))．最初の層の前には位置エンベディング (Positional Embedding) を適用し，モデルにレシピテキストの位置情報も学習してもらう．次元数　= 512，2 つの層はそれぞれ 4 つの attention head が付く．文のエンベディングは Transformer エンコーダー最終層の出力の和の平均となる．タイトルエンベディング $4_{CC;}$ は，このように得られる．

3.1.2　調理手順・成分レベルのエンコーディング

レシピテキストにあるタイトルはすべて一つに定まる一方，サンプルに含まれる調理手順・成分の文が複数含まれる．本研究は，H-T [13] と同様に二つの Transformer(図 2(a)) をもつ，階層的 Transformer(HTR)(図 2(b)) を調理手順・成分の学習に導入する．調理手順と成分の処理は同じで，ここでは便宜上，成分を例として本研究において複数の文をもつサンプルの処理手法を紹介する．

レシピテキストの成分として与えられた $"$ 個の文に対して，最初の Transformer $()'_{!=1}$) はそれを先に $"$ 個のサイズが固定されたエンベディング $4_{8="}0$-$4_{8="}1$-$\cdots$-$4_{8="}{}_{"}$ に変換させ，二つ目の Transformer$()'_{!=2}$) は変換されたすべての成分の文のエンベディングを一つ統一されたエンベディング $4_{8="}$ に変換させる．調理手順のエンベディング $4_{8=B}$ も上述の方法で得る．次は，得られた三つの成分のエンベディング $4_{CC;}$-$4_{8="}$-$4_{8=B}$ を，一つ統合されたレシピテキストエンベディング $'$ にエンコードす

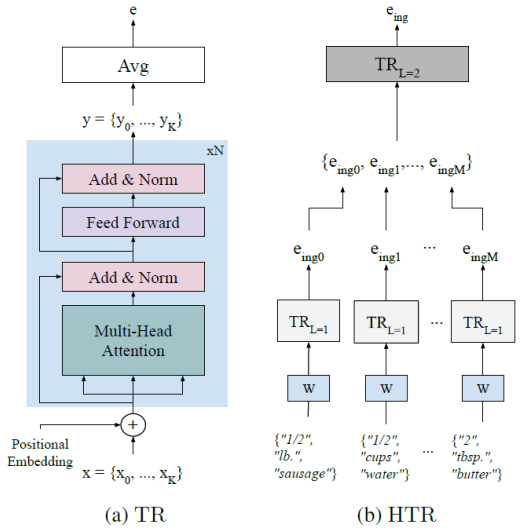

図 2　Transformer と階層的 Transformer([13] より引用)

る手法を紹介する．

3.1.3　タイトル・成分・調理手順の間の補完関係

本研究では信頼性のあるレシピテキストエンベディング ′ を得るために，H-T [13] と同様に自己教師ありレシピテキスト損失のトリプレットロス $!_{A42}$ を導入する．この損失の導入により，レシピ画像がない text-only のサンプルも利用でき，レシピテキストに含まれるタイトル・成分・調理手順の間の補完的関係を学習することが可能になる．対象のエンベディングは三つあるため，双方向のトリプレットロス関数 (Bi-directional Triplet Loss Function) を導入する．

$$!'_{18}(8\text{-}9) = [2(4_0^{(8)}\text{-}4_1^{(9)}) - 2(4_0^{(8)}\text{-}4_1^{(8)}) + U]_+ + [2(4_1^{(8)}\text{-}4_0^{(9)}) - 2(4_1^{(8)}\text{-}4_0^{(8)}) + U]_+ \quad (2)$$

ここでは，$4_0\text{-}4_1$ は異なるエンベディング集合で，上付き文字の要素は互いにポジティブ $(4_0^{(8)}\text{-}4_1^{(8)})$，そうではない要素は互いにネガティブ $(4_0^{(8)}\text{-}4_1^{(9)}$ や $4_1^{(8)}\text{-}4_0^{(9)})$．また，$2[\cdot]$ はコサイン類似度，$[I]_+ = <0G(0\text{-}I)$，マージン U は 0.3 に設定する．所属集合を表す下付き文字は $0\text{-}1 \in \{CC;\text{-}8=\text{"-}8=B\}$．式 2 をさらに発展させ，以下の通りにタイトル・成分・調理手順の任意の二つに適用する，一つバッチ内のトリプレットロス関数を定式できる．

$$!_{18}(0^{(8)}\text{-}1^{(8)}) = \frac{1}{} \tilde{O}_{9=0} !'_{18}(8\text{-}9)X(8\text{-}9)\text{-} \quad (3)$$

$$X(8\text{-}9) = \begin{cases} 0 & \text{if } 8 = 9 \\ 1\text{-} & \text{otherwise} \end{cases}$$

はバッチサイズで，X は相関関数を表し，$0\text{-}1 \in \{CC;\text{-}8=\text{"-}8=B\}$．

学習時にすべてのエンベディングが等しくなることを防ぐために，H-T [13] と同様に本研究では線形レイヤーを利用しエンベディング間の写像を適用してから損失関数を計算する．図 3 の示した通りに，線形レイヤーを用いて目標エンベディング空間に適用してからロスを計算する．$"_{0-i1}(\cdot)$ は，集合 0 の

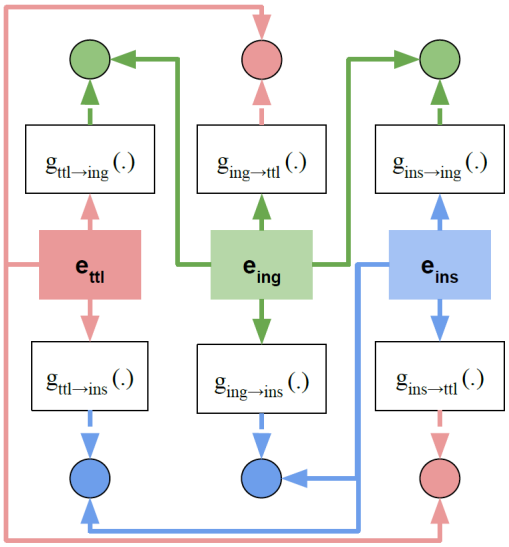

図 3　タイトル・成分・調理手順エンベディング間の線形変換と自己教師あり学習ロス計算 ([13] より引用)，異なる色のドットは各レシピテキスト成分のロス項目を示す

空間から集合 1 の空間への線形変換である．例えば，$4_{CC;}$ が $"_{CC;-i8="}(\cdot)$ を用いてタイトルエンベディングが成分エンベディング空間における投影 $4_{CC;-i8="}$ に変換してから，成分エンベディング $4_{8="}$ との損失の式は 4 の通りに定式化できる．

$$!'_{A42}(8="\text{-}CC;) = !_{18}(4_{CC;}^{(8)}\text{-}4_{CC;\to 8="}^{(8)}) \quad (4)$$

式 4 を一般化すると，以下のようになる:

$$!'_{A42}(0\text{-}1) = !_{18}(4_0^{(8)}\text{-}4_{1\to 0}^{(8)}) \quad (5)$$

ここでは，$0\text{-}1$ はすべての異なるレシピテキストの成分（タイトル，成分，調理手順）のエンベディングを取り，コサイン類似度の距離学習を行う三つのエンベディング空間（タイトル，成分，調理手順）があるため，その変換は合計 6 種類あり (図 3)，ロスも 6 種類計算する．式 6 にようにまとめられる：

$$!_{A42} = \frac{1}{6} \tilde{O}_{0} \tilde{O}_{1} !_{18}(0\text{-}1)X(0\text{-}1)\text{-} \quad (6)$$

$$X(8\text{-}9) = \begin{cases} 0 & \text{if } 8 = 9 \\ 1\text{-} & \text{otherwise} \end{cases}$$

繰り返しになるが，タイトル・成分・調理手順の三つの空間を対象とするため，$0\text{-}1 \in \{CC;\text{-}8=\text{"-}8=B\}$．合計 6 種類のロスが計算されるため，レシピテキストロスの式 6 は 6 で割る．この自己教師あり学習ロスで学習を行った後，学習された三つのエンベディンググ $4_{CC;}\text{-}4_{8="}\text{-}4_{8=B}$ をエンコーダー $_{<A"}$ を通して，最終のレシピテキストエンベディング ′ を得る．

3.2　画像エンコーダー

画像エンコーダー $_+$ を用いて，レシピ画像を画像ーテキスト共有埋め込み空間への写像関数を学習することが目的とする．本研究では，最新研究の ViT [3] を画像エンコーダーに利用し

た．具体的には，同じく [2] の事前学習済みの ViT ベースサイズモデルを利用した．また，同時に，ImageNet [2] で事前学習済みの ResNet-50 [8] を利用する実験も行った．ResNet-50 の分類器直前のレイヤーの次元数 = 1024 の出力を画像埋め込みとして利用した．

3.3 モダリティ間の埋め込み学習

本研究では，モダリティ間のギャップを緩和するため，ACME [18] と同様にモダリティ間の融合に敵対的学習を導入し，モダリティ間のギャップ問題を緩和する．

ここでは，テキストと画像の二つのモダリティからの埋め込みをよりよく融合するために弁別器を導入する．この弁別器は，与えられた埋め込みの元は画像とテキストのどちらかを判別するものである．弁別器が与えられた埋め込みの元が判別できないまで学習させることで，画像・テキストからの埋め込み ((i) (t)) をよりよく融合できる．このモダリティを融合 (Modality Alignment) する損失関数は式 7 のように定義できる．

$$! = \mathbb{E}_{\mathbf{i}\sim ?(8)}[\log(1 - (\mathbf{i}))] + \mathbb{E}_{\mathbf{t}\sim ?(C)}[\log(1 - (\mathbf{t}))] \quad (7)$$

3.4 画像・テキストクロスモーダル検索の距離学習

以上の手順を踏まえて, 画像・テキスト埋め込み + = (i) - ′ = (t) を得られた．サンプル間の検索の距離学習のためのトリプレットロス [15] を式 8 のように導入する．マージン $U = 0{\cdot}3$, $3(\cdot)$ はユークリッド距離で，$[G]_+ = <0G(G-0)$ である．

$$!_{A4C} = \tilde{O}_{+}[3(+_{0}-{}'_{?}) - 3(+_{0}-{}'_{=}) + U]_{+} + \tilde{O}_{'}[3({}'_{0}-+_{?}) - 3({}'_{0}-+_{=}) + U]_{+} \quad (8)$$

3.5 埋め込みと元情報の一致性

ACME [18] により時折，検索には有用なエンベディングが得られたが，元のモダリティにおける固有情報が失われたことがある．この状況を防ぐため，ACME [18] と同様にエンベディングの一致性を持つロスを導入する．このロスは，レシピエンベディング一致性ロス $!_{CAO=B_r}$ と，画像エンベディング一致性ロス $!_{CAO=B_i}$ の二つの部分をまとめたロスとなる：

$$!_{CAO=B} = !_{CAO=B_r} + !_{CAO=B_i} \quad (9)$$

3.5.1 テキスト埋め込みからのレシピ画像生成

レシピエンベディング一致性ロス $!_{CAO=B_r}$ は，二つのロス $!_{A28}$ - $!_{2;B_{r2i}}$ を取り組んだロスである．$!_{A28}$ は，生成画像がなるべく本物料理に見えるために導入したロスである．GAN [5] を用いて，FC レイヤーを通ったレシピエンベディング (**FC**( ) (t))) から画像を生成する．式 10 のように定式化できる．$!_{2;B_{r2i}}$ は，生成画像が目的の「正しい料理画像」を制限するために導入するロスである．ここでは料理タイトル分類器を適用し，クロスエントロピーロス $!_{2;B_{r2i}}$ を導入する．

$$!_{A28} = \mathbb{E}_{\mathbf{i}\sim ?(8)}[\log(1 - {}_{A28}(\mathbf{i}))] + \mathbb{E}_{\mathbf{t}\sim ?(C)}[\log(1 - {}_{A28}((\mathbf{FC}( ) (\mathbf{t})))))] \quad (10)$$

以上の二つのロスをまとめると，最終のレシピエンベディングの一致性を確保するためのロスである：

$$!_{CAO=B_r} = !_{A28} + !_{2;B_{r2i}} \quad (11)$$

3.5.2 画像埋め込みからのレシピ成分予測

画像エンベディング一致性ロス $!_{CAO=B_i}$ も同様に，二つのロス $!_{82A}$ - $!_{2;B_{i2r}}$ を取り組んだロスである．$!_{82A}$ は，画像エンベディングを用いて成分予測のロスである．従来の研究 [1], [18] では，マルチラベル分類で料理の成分を予測する手法がある．ここでは，ACME [18] と同様に，レシピ画像埋め込 + みをマルチラベルネットワークに入れることで，予測した成分を表す次元数 4102 の one-hot ベクトル (4102 次元は 4102 種類の成分を表す) を得り，このロスを $!_{82A}$ とする．$!_{2;B_{i2r}}$ は，料理のカテゴリーを正しく得るためのクロスエントロピーロスである．以上により，最終の画像埋め込みの一致性を確保するためのロスが定式化できる：

$$!_{CAO=B_i} = !_{82A} + !_{2;B_{i2r}} \quad (12)$$

4. 実　　験

ここでは，提案手法のレシピクロスモーダル検索フレームワークの有効性を検証するための実験を紹介する．検索精度の確認のほか，敵対的学習による生成画像の結果も紹介する．

4.1 データセット

データセットは，前述の通りで Recipe1M [14] を利用する．我々は学習データ，検証データ，テストデータにはそれぞれ 238,999，51,119，51,303 の画像-レシピペアを利用する．また，レシピの自己教師あり学習の適用のため，データセットの残りの 482,231 の画像が含まれないレシピだけのサンプルを学習に使用する．

4.2 学 習 設 定

学習の最適化目標は，式 1 を最小化することである．実験では，$\_1$ を 0.05，$\_2$ を 0.005，$\_3$ を 0.002 に設定した．また，学習率・バッチサイズをそれぞれ 256，10^{-4} に設定し，Adam [11] optimizer を適用して学習を行う．学習時は，1k サイズにおけ

図 4　検索例 (Ours+ViT).

表 1　検索性能比較．太字の結果が最上位結果で，アンダーラインがついている結果が 2 位

	1k								10k							
	Image-to-Recipe				Recipe-to-Image				Image-to-Recipe				Recipe-to-Image			
	medR	R@1	R@5	R@10	medR	R@1	R@5	R@10	medR	R@1	R@5	R@10	medR	R@1	R@5	R@10
JE [14]	5.2	24.0	51.0	65.0	5.1	25.0	52.0	65.0	41.9	-	-	-	39.2	-	-	-
R2GAN [20]	2	39.1	71	81.7	2	40.6	72.6	83.3	13.9	13.5	33.5	44.9	12.6	14.2	35.0	46.8
MCEN [4]	2	48.2	75.8	83.6	1.9	48.4	76.1	83.7	7.2	20.3	43.3	54.4	6.6	21.4	44.3	55.2
ACME [18]	1	51.8	80.2	87.5	1	52.8	80.2	87.6	6.7	22.9	46.8	57.9	6	24.4	47.9	59.0
SCAN [19]	1	54.0	81.7	88.8	1	54.9	81.9	89.0	5.9	23.7	49.3	60.6	5.1	25.3	50.6	61.6
IMHF [12]	1	53.2	80.7	87.6	1	54.1	82.4	88.2	6.2	23.4	48.2	58.4	5.8	24.9	48.3	59.4
RDEGAN [16]	1	59.4	81.0	87.4	1	61.2	81.0	87.2	3.5	36.0	56.1	64.4	3	38.2	57.7	65.8
H-T [13]	1	60.0	87.6	92.9	1	60.3	87.6	93.2	4	27.9	56.4	68.1	4	28.3	56.5	68.1
X-MRS [6]	1	64.0	88.3	92.6	1	63.9	87.6	92.6	3	32.9	60.6	71.2	3	33	60.4	70.7
Ours	**1**	66.5	89.1	93.3	**1**	66.2	89.3	93.6	3	36.4	63.6	73.8	3	36.6	63.6	73.7
Ours+ViT	**1**	**73.0**	**91.8**	**95.2**	**1**	**72.9**	**91.9**	**95.2**	**2**	**44.3**	**70.9**	**79.7**	**2**	**44.6**	**70.8**	**79.5**

表 2　バッチサイズが提案手法における影響の比較（バックボーン：ResNet50)

test set size		Image-to-Recipe				Recipe-to-Image			
	batch size	medR	R@1	R@5	R@10	medR	R@1	R@5	R@10
1k	64	1	61.6	87.8	92.4	1	62.3	87.8	92.8
	128	1	65.0	89.0	93.3	1	65.4	89.1	93.5
	256	1	66.5	89.1	93.3	1	66.2	89.3	93.6
10k	64	3.7	31.2	58.3	69.2	3.4	31.9	58.7	69.5
	128	3.0	35.0	62.6	73.6	3.0	35.4	62.6	73.1
	256	3.0	36.4	63.6	73.8	3.0	36.6	63.6	73.7

表 3　バッチサイズが提案手法における影響の比較（バックボーン：ViT-B)

test set size		Image-to-Recipe				Recipe-to-Image			
	batch size	medR	R@1	R@5	R@10	medR	R@1	R@5	R@10
1k	64	1.0	64.0	88.0	92.8	1.0	64.3	88.0	92.8
	128	1.0	70.8	91.3	94.9	1.0	70.8	91.4	95.0
	256	1.0	73.0	91.8	95.2	1.0	72.9	91.9	95.2
10k	64	3.0	34.0	60.8	71.1	3.0	34.0	60.5	70.8
	128	2.0	41.3	68.3	77.5	2.0	41.5	68.5	77.7
	256	2.0	44.3	70.9	79.7	2.0	44.6	70.8	79.5

る R @ 1 を基準として，エポックの最後に best model を更新する．また，本研究は 2 種類の画像エンコーダー，ResNet50 [8] と ViT [3] を利用し，その結果を比較する．

4.3　評 価 指 標

以前の研究に従って，本研究は median rank (medR)， Recall@{1,5,10}の基準で検索精度を評価する．また，テストサイズを 1k，10k に分けて結果をまとめる．検索精度は，テストデータから 5 回ランダムにデータを選択し，得られた結果の平均を実験結果として報告する．

4.4　検索精度の比較

実験結果は表 1 の示した通り，提案手法が一番優れた結果を得られた．Ours は我々の提案手法の画像エンコーダーが ResNet50 である場合で，Ours+ViT は提案手法の画像エンコーダーが ViT のベースパッチを使った場合である．二つの実験結果はともに優れた結果が得られた．特に ViT を適用した場合は，従来の手法を大幅に上回った検索精度を得られた．図 4 は提案手法の検索例である．上位にある画像は，少なくとも似たような外観をもつ料理である．

4.5　画 像 生 成

図 5 は生成画像の一部の例である．生成画像が，ベースラインの ACME と比べて，質のよりよい画像が生成できたと考える．表 4 は，オープンソースコード[1]を用いた FID スコア [9] の比較結果である．FID スコアは，生成画像の質を評価するための基準であり，低ければ低いほど生成画像の質が良いと示される．結果の示した通りに，提案手法もすぐれた結果が得られた．

表 4　FID score の比較

Method	FID
ACME [18]	30.7
X-MRS [6]	28.6
Ours	**23.3**
Ours + ViT	27.1

4.6　バッチサイズがタスクにおける影響について

我々はレシピテキストの学習と，検索するための距離学習の二箇所のところにトリプレットロスを適用している．以上の二つはすべて一つのバッチ内で距離学習を行うので，バッチサイズが増えると，モデルの距離学習もより適切に行えると考える．表 2，3 は，バッチサイズを変えて記録した実験結果である．画像エンコーダーが ResNet50 か ViT であるかにかかわらず，バッチサイズが増えるにつれて検索精度も上がった．特に，ViT での精度改善が ResNet50 の 2 倍となった．

また，一番優れた結果を得た実験設定 (バッチサイズ 256，画像エンコーダーバックボーン:ViT ベースパッチ) では，TITAN RTX (24GB)×4，合計で 96GB のメモリーを利用した．一番軽量な実験設定 (バッチサイズ 64，画像エンコーダーバックボー

(注1)：https://github.com/mseitzer/pytorch-fid